AFRL-RI-RS-TR-2008-74
Final Technical Report
March 2008

TARGETED INFORMATION DISSEMINATION

Quantum Leap Innovations, Inc.

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

NOTICE AND SIGNATURE PAGE

AFRL-RI-RS-TR-2008-74 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/	/s/
NANCY A. ROBERTS Work Unit Manager	JOSEPH CAMERA, Chief Information & Intelligence Exploitation Division Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
MAR 2008	Final	Mar 05 – Dec 07

4. TITLE AND SUBTITLE

TARGETED INFORMATION DISSEMINATION

5a. CONTRACT NUMBER
FA8750-05-C-0104

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62702F

6. AUTHOR(S)

Srikanth Kallurkar

5d. PROJECT NUMBER
558E

5e. TASK NUMBER
05

5f. WORK UNIT NUMBER
B1

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Quantum Leap Innovations, Inc.
3 Innovation Way, Ste 100
Newark DE 19711-5456

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RIED
525 Brooks Rd
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2008-74

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA# WPAFB 08-0669

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Quantum Leap Innovations (QLI) developed a Targeted Information Dissemination (TID) system for rapid gathering and dissemination of the right information to the right people at the right time. The TID user interface shows tasks of an analyst. A hierarchical view of interests learned over a period of time is shown for each task. A table displays documents filtered-in by the user agent. The filtering is based on an interest profile that the agent manages on behalf of the user. The user can view and change the degree of filtering, document relevance and the interests related to task at any time. QLI focused their system to derive an early warning system (EWS) posed by a potential pandemic influenza (PI) episode, but the technology will be broadly applicable and configurable as an EWS for any future biological incident.

15. SUBJECT TERMS
Information Dissemination, virtual interest group, interest learning, information routing, collaboration, interest profile, structured peer-to-peer network

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Nancy A. Roberts
U	U	U	UL	50	19b. TELEPHONE NUMBER (*Include area code*) N/A

Table of Contents

LIST OF FIGURES

Executive Summary

Quantum Leap Innovations (QLI) researched and developed for the intelligence community, software architecture for rapid gathering and dissemination of the right information to the right people at the right time.

The underlying cause of all intelligence "failures" is that we have not extracted, in a timely manner, the right knowledge from the massive amounts of information available. History is replete with examples where we have been surprised or disadvantaged, even when dealing with relatively precise information from sophisticated technical sensors. As illustrated daily in the Global War on Terrorism, the difficulty of intelligence analysis today is even more difficult, due to increased dependence on human sources, which often can be deceptive, incomplete, inaccurate, contradictory, and/or irrelevant. Once the often overwhelming amount of available data is processed, the intelligence analyst is faced with an even more daunting challenge – determining what it all means. For this task, the intelligence analyst must utilize their experience and intuition, but most importantly, collaboration with other analysts.

The Targeted Information Dissemination (TID) system being developed by (QLI) will significantly enhance the capabilities of intelligence analysts by providing effective and efficient transmission of relevant all-source intelligence information, as well as identifying potential collaboration partners.

The TID user interface shows tasks of an analyst. A hierarchical view of interests learned over a period of time is shown for each task. A table displays documents filtered-in by the user agent. The filtering is based on an interest profile that the agent manages

on behalf of the user. The analyst can view and change the degree of filtering, document relevance and the interests related to task at any time.

In addition, QLI leveraged this work and domain expertise in developing intelligent computing solutions to address an as yet unmet need to derive an early warning system (EWS) for a biological incident. The focus for the EWS was the uncertain but urgent threat that is posed by a potential pandemic influenza(PI) episode, but the technology will be broadly applicable and configurable as an EWS for any future biological incident.

Perfect knowledge is likely unattainable, but Quantum Leap Innovations' Targeted Information Dissemination (TID) system will give intelligence analysts a much better opportunity of extracting the right knowledge from all available information and producing finished intelligence to policy and decision makers that is timely, accurate, and actionable.

The system uses open-source data but is based on network-centric and service-oriented architecture providing for use by USNORTHCOM (or other parties) on LAN, WAN or Web-based networks (including where applicable CLASSIFIED). The EWS provides a multi-phase capability with user specified triggers and thresholds to allow for continuous operation without the user being 'swamped' by irrelevant data.

1. Goals and Objectives

QLI's goal (Figure 1) was to provide for rapid information gathering and dissemination to intelligence analysts (users) to enable accurate and timely execution of intelligence tasks.

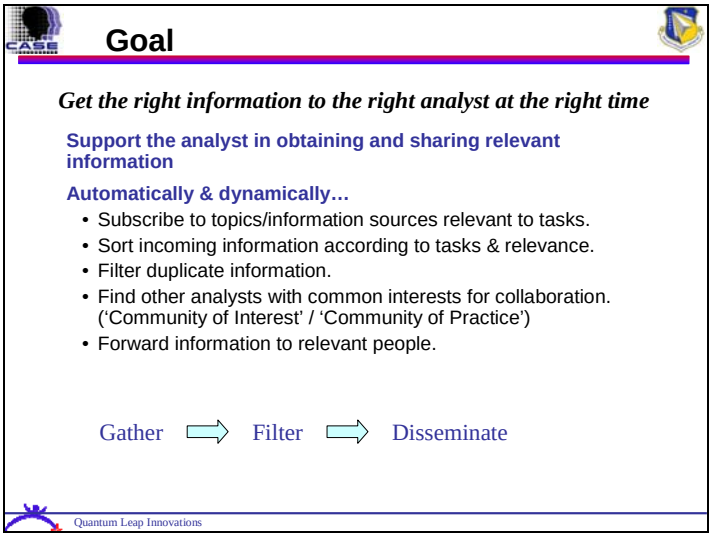


Figure 1: TID Goals

The principal objectives were to develop:

- Mechanisms to represent, analyze and evolve user interests
- Mechanisms to share information within a large set of distributed users

A user profile describes user's information interests that are relevant to user tasks. The profile provides for identification of users with common interests. Users with common interests are grouped in Virtual Interest Group (VIG) to allow for sharing and disseminating relevant information.

2. Overview

TID, represented in Figure 2, provides software mechanisms to automatically store & maintain analyst interests

- Based on analyst's tasks, activities & incoming information
- Analyst can inspect & modify profile at any time
- Analysts with common interests form Virtual Interest Group (VIG)

It provides for sharing and disseminating information to the relevant users.

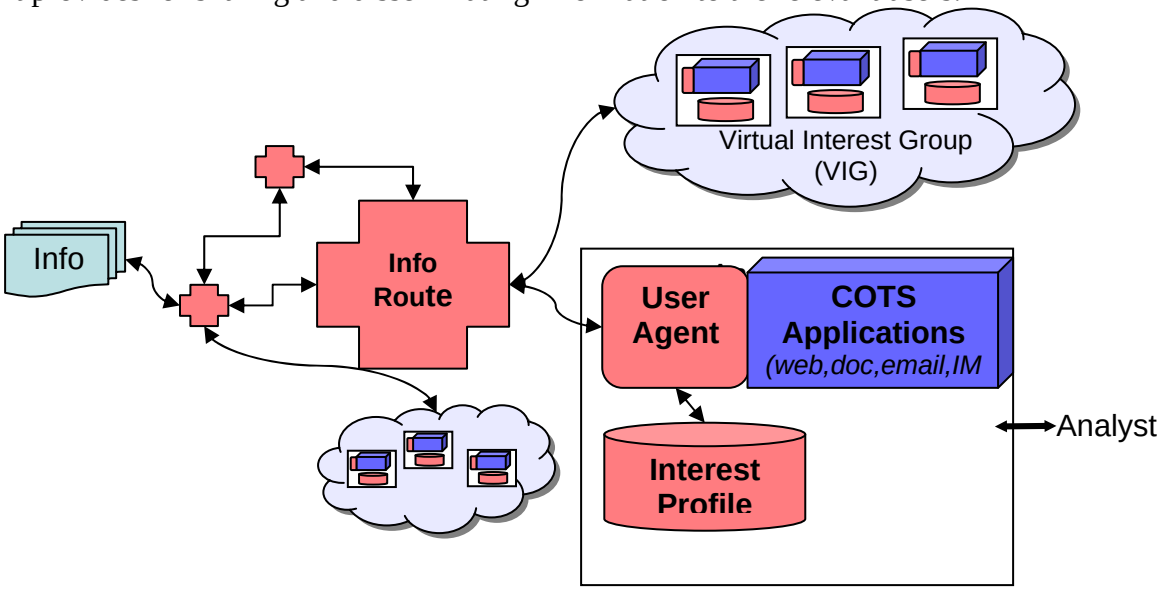


Figure 2: TID System Overview. Information flows in the system through a set of routers via VIGs to user agents. Agents filter, process and present information to users via COTS applications.

The TID architecture specifies two main components for achieving the objectives. These two components are the Information Network and the User Agent. The Information Network is composed of two layers. The first layer is made up of Information Routers that are peers in a structured Peer-to-Peer routing network. The information router's main function is to cooperatively route information to the most suitable destination router. The second layer is comprised of Virtual Interest Groups (VIGs).

2.1. Information Interest and Profile

Without loss of generality, TID system defines information interest as salient "features" describing or being about information. The feature representation can be by 1) topic titles in a topic taxonomy, 2) text summary of a document or a document collection, 3) centroids of document collections, 4) dominant concepts in a Singular Value Decomposition vector, 5) metadata of a SQL database, 6) concepts or classes from a domain specific ontology, and etc. The selection of salient features is determined by the granularity of an interest. For example, information interest "Iraq" can be composed with specific topic titles from a *country* and *events* ontology. The salient feature set can include *neighbors*, *weather* and *demographics* of *Iraq* from the *country* ontology while *bombings* and *music shows* in *Iraq* from the *events* ontology. Thus, the interest "Iraq" has

salient features *neighbors, weather, demographics, bombings* and *music shows* related to Iraq. A simpler feature set of the interest "Iraq" could be the country name "Iraq". The first case shows that the user is interested in very specific features while in the second example the user is interested in "anything" related to Iraq. Generally, features are descriptors of a user's information interests and interests can be composed of several low level or basic features. The system utilizes representation specific mechanisms to decompose user interests into features or sub-interests. Such feature creations provide a basis for routing information that may not directly related to the user's top level interest but relevant to a feature (or sub-interest) that the user may be unaware of.

The user information interest profile maintains such features or sub-interests of a user. The presence of an interest in a user profile indicates that the user is "interested" in information relevant to that *interest*. A user can be both a producer and consumer of such relevant information. An information profile can comprise many such interests relating to the myriad of information needs of a user.

2.2. User Agent

A user is represented in TID by a *User Agent* (hereafter referred to as *agent*) that resides on the user's workstation (or any other computing device) and allows the user to interact with the TID system. The agent creates and manages a user information profile through profile management behaviors. Profile management is driven by user inputs and automated learning mechanisms about user needs and context awareness. The agent manages a knowledge base that is relevant and current with respect to the profile. It uses various information retrieval, filtering and data mining mechanisms to manage the profile and maintains the best notion of information relevance and novelty. An agent manages multiple profiles depending upon the user's information needs. For each interest in the profile, the agent manages memberships to the various Virtual Interest Groups.

2.3. Virtual Interest Group

A Virtual Interest Group (VIG) is a logical grouping of users that have common or shared interests. A VIG corresponds to a feature of an interest. Information relevant to the group is received by a VIG representative and disseminated to the members of the VIG.

2.4. Information Network

The Information Network provides vital interconnection capabilities in TID system. It allows for creation and management of the VIGs and the publication and routing of information through information routers. The agent members of the VIGs interface with the information network to publish and consume information. A VIG forms a part of the information network. Figure 3 depicts an overview of the TID System. Various users are represented by their agents. An agent resides on the user workstation to interact with the user. It creates and manages profiles and memberships to relevant VIGs. Information relevant to a VIG (i.e. to the corresponding interest of the VIG) is disseminated to members. The agent performs appropriate filtering before providing the user with the information.

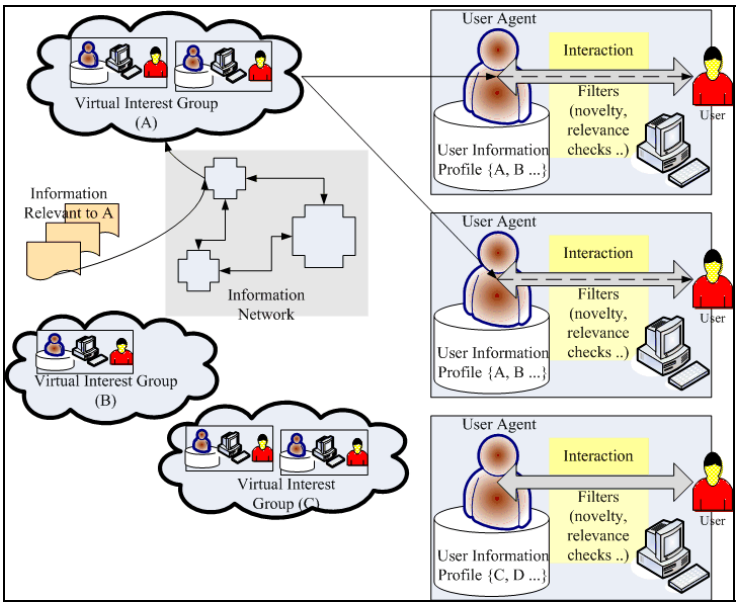


Figure 3: Overview of TID System. Users are represented by User Agents residing on user PCs. The agents manage memberships to various VIGs. The information network disseminated information to the needful users through routers and VIGs.

3. System Architecture

The TID system architecture is modular to allow the development of pluggable components. The architecture has two high level components – Information Network and User Agent. The following sections describe their subcomponents.

3.1. Information Network

Unique Interests	Users in a TID System				
	$User_1$	$User_2$	$User_3$	…	$User_m$
I_1	X	X			
I_2			X		
I_3		X			
…					
I_n	X				X

The TID information network provides for sending and receiving of information relevant to the users based on their information interests. This requires a mechanism of identifying or discovering users with a specific interest.

Figure 4: Global view of unique interests of users in a TID application.

A mechanism is needed that maps interests to pertinent users.

Figure 4 depicts a hypothetical global view of unique interests that the users may have in a TID application. Such a view can be highly transient when users add, change, delete, join or leave interests. The discovery mechanism must be able to manage such a view to provide for effective and efficient information dissemination. The standard approach for discovery has been a centralized repository or a directory facilitator that can be queried or looked up. So, in a centralized approach, the TID system maintains a centralized repository of interests and a map of users pertinent to the interest. Information would be resolved to its interests and lead to a look up of corresponding users. Indeed, communities of interests that span the Global Interest use a similar mechanism where users subscribe to the server through a web interface and publish and receive relevant information. The solution is simple to implement and use but poses the problem of a central point of failure, bottlenecking, commitment of large resources to maintain the repository and a non-scalable solution to growth in information volume and complexity. A newer approach is based upon distributed implementation of the global view of interests and corresponding users. The TID system architecture provides for such a distributed mechanism of discovering users for a specific information interest to alleviate some of problems associated with a centralized approach and is described below.

The TID information network (Figure 5) has two layers. The first layer comprises Information Routers (hereafter called routers) interconnected by a single logical shared network. The routers represent a single interest on the shared network to publish, receive and route information relevant to the interest. The second layer comprises VIGs for each of the interests identified by the TID application. A VIG is populated by user agents that have the same VIG "interest". The VIG has its own internal logical network for disseminating information to the member agents. The VIG internal network is independent of the first layer shared network. The two layers together provide for effective and efficient information routing and filtering so that the users receive information relevant to their interests. There exists a One-to-One mapping between the

routers in the first layer and the VIGs in the second layer. That is, each VIG has a dedicated router and each router represents a single interest and the associated VIG.

3.2. Information Routing

Information Routing refers to a mechanism of distributing information on a network. An application of information routing is in distributing a query for an information need in a network of nodes that share information. The purpose of routing the query is to obtain relevant information from the nodes in the network. The most common distribution mechanism is "flooding" where the query is flooded to *all* nodes in a network. The "flooded" nodes compare the received query with their shared information and send the results back to the "flooder". The main disadvantage with this mechanism is that nodes that do not have information relevant to the query will receive the query and waste their computing resources on processing the query. In addition, message traffic due to flooding increases the load on the system. A different approach to routing information is that the nodes share metadata about their information content. Each node has access to the shared metadata about every other node. A query can be compared to the shared metadata to obtain "hints" about the actual information stored at the nodes. The best matching nodes are then selected and queried thereby greatly reducing the message traffic load on the system. The main problem in this approach is to have a efficient mechanism of sharing metadata about each node. A simple approach is to have a centralized repository of metadata updated by the nodes individually. This approach poses the same problems of any centralized approach of single point of failure and commitment of large resources and scalability to the growth of information volume and nodes in the application. Another approach is that each node stores metadata about each of the other nodes in the network. This approach works well only for a small number of nodes and when the nodes do not update metadata frequently. A third and increasingly popular approach is that each node stores metadata about a small subset of nodes in the network. In this approach, a query is compared by the node with its set of stored metadata. The best matched nodes are sent the query to obtain the results. If metadata comparisons do not provide a suitable match, the query is forwarded to a set of "neighboring" nodes so that each of those nodes can compare the query with their set of stored metadata. The disadvantage of this approach is that an ad hoc mechanism of sharing metadata does not guarantee complete coverage of the metadata about nodes in the network. These deficiencies have led to research and development of establishing structure in routing networks. The TID system architecture uses a similar structured approach of discovering routers in the network. The interconnection network is a Peer-to-Peer (P2P) structured network and applies to the first layer in the information network component.

3.2.1. Structured P2P Network

A structured P2P network consists of peers that are addressed by a specific mechanism such that each peer address pertains to a specific position on the overlay network. An overlay is a logical topology of the network. For example, a circle can be logical topology where the peers represent distinct points on the circumference of a logical circle. The term overlay simply refers to such a logical network because the peers form an overlay over actual physical network (such as a LAN or the Global Internet). The advantage of a structured network is that with a known addressing mechanism, relevant

peers can be quickly identified and thus reduces message traffic seen in broadcast unstructured P2P networks. Most commercial P2P networks are unstructured networks such as file-sharing networks Kazaa and Gnutella. The general framework of a structured network is that each peer maintains a small set of neighbors known as leaf nodes. The network is maintained by the updating the leaf nodes when nodes join or leave the network. The maintenance of leaf nodes by each peer provides for compete connectivity of the network. Each peer also maintains a routing table which contains peer addresses. The composition of the table is based upon the peer addressing mechanism.

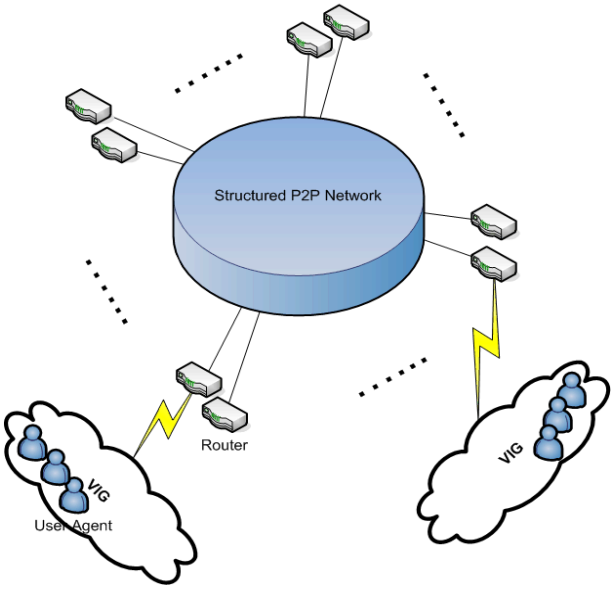


Figure 5: Overview of information network. User agents are members of VIGs. A router interfaces with a VIG and the P2P network. The routers form the 1st layer and VIGs form the second layer.

The composition itself provides for efficient routing in bounded time. Addressing mechanisms can differ by application needs. Since data or file sharing is a prime application for P2P networks, many algorithms derive addresses for peers based on their information content. Thus, a query about a particular information need can be routed to those peers whose address is based on information relevant to the need. Furthermore, the same query routing mechanism can be applied to *information routing* to disseminate information to the pertinent users.

3.3. Information Network Architecture

The main issue with a structured network is that it trades effectiveness of information need satisfaction with message routing efficiency. Most structured topologies do not perform at the same message routing efficiency when the information being shared is high dimensional. To counter this deficiency, the TID information network (architecture shore in Figure 6) is divided into two distinct layers. The first layer routers are interconnected by a “structured” Peer-to-Peer (P2P) network with each router having a single peer interface to the overlay network. Each router is also associated with a single VIG. The routers provide a first coarse grained filter to incoming information. The second layer VIGs provide finer grained filter through the member agents so that information most specific to the users can be channeled to the users.

3.3.1. Information Router

An information router's objective is to channel information relevant to the interest into the VIG. A router establishes its P2P overlay address based on metadata about the interest. To route information to pertinent routers, the information is resolved to its constituent interests. Each interest is then resolved to an overlay address using the same router addressing mechanism. The result of this resolution process is a list of routers that have the most suitable metadata to the resolved interests (i.e. closest matching addresses).

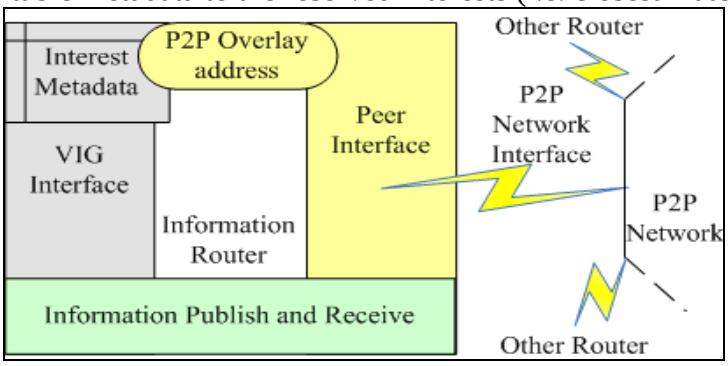


Figure 6: Architecture of an Information Router. The router has interfaces to the VIG and the P2P network. Its address on the P2P network is derived from interest metadata.

The message is then routed to these routers. A receiving router disseminates information to the members of the VIG. A router comprises the following modules/interfaces - Peer Interface (Address Resolution), VIG Interface and Information Publish and Receive.

3.4. Virtual Interest Group Architecture

A Virtual Interest Group's objective is to publish and receive information on behalf of the member agents. A VIG is associated with a router that allows the VIG to publish and receive information relevant to its interest. The TID architecture (Figure 7) specifies a single VIG for an interest such that all agents join the VIG for the particular interest. A VIG has a leader, called Supernode that provides an interface with the router. The TID architecture does not specific who, how or how many Supernodes be nominated. An example of nominating a Supernode is that the first user agent to establish a VIG for an interest becomes the Supernode. The Supernode publishes metadata to the router and provides an agent interface to the other members of the group. Information is disseminated in the VIG through this interface.

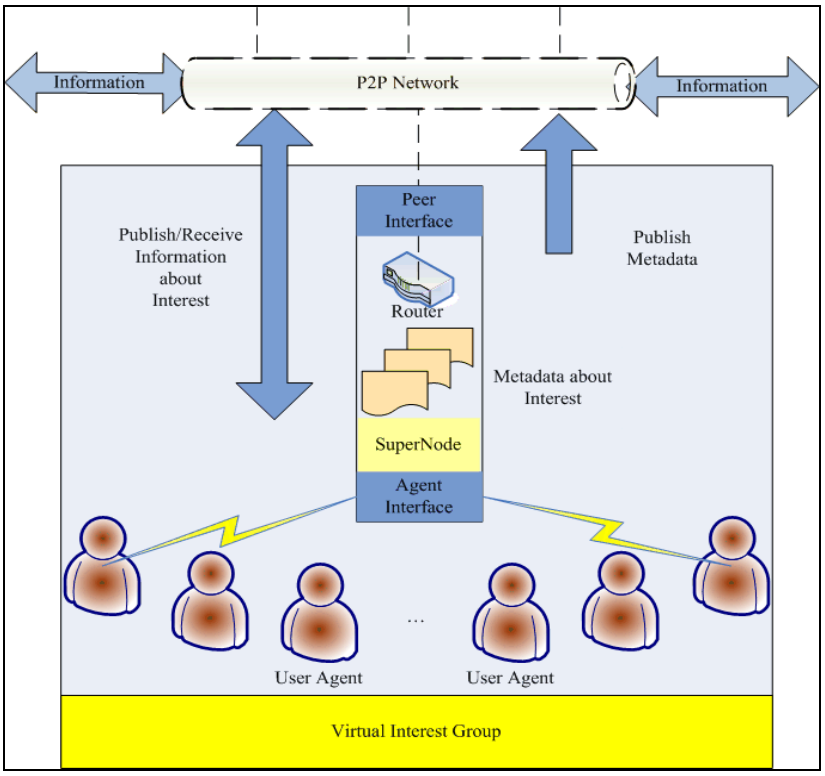


Figure 7: Architecture of a VIG. A VIG comprises member agents. It is represented by a Supernode that has agent interface to the other agents and router interface to publish and receive information the first layer. The Supernode also manages the VIG metadata.

3.5. User Agent

A User Agent is a software entity that represents a user in the TID architecture (Figure 8). The agent's main function is to gather information that can assist the user in performing his or her tasks. The agent is comprised of three main modules – User Profile Management, VIG Management, and User Interface Management. The agent creates a user profile and stores it locally. It provides for a user input interface for the user to manipulate the profile. The agent maintains a knowledge base of the information relevant to the user and displays relevant information to the user through appropriate displays. The profile management provides the agent with interests for managing appropriate VIG memberships. The agent finds appropriate VIGs in the Information Network through an Information Router. Changes in the profile result in changes in a user's VIG memberships.

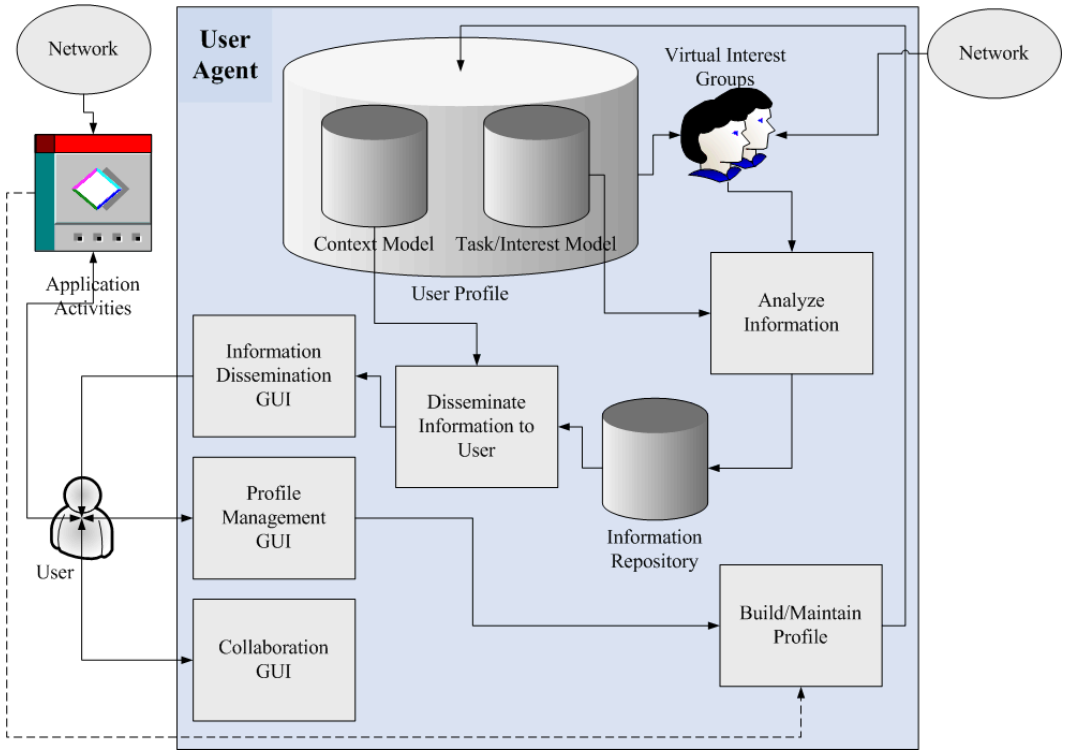


Figure 8: User Agent in TID Generalized Architecture

Figure 6 shows User Agent architecture and its interactions with the various components. In the next section we describe a generalized architecture of a User Agent's

3.6. User Profile Management module

A User Profile (Figure 9) is comprised of task, interest, and context models associated with the user. So far, we have focused primarily on the task and interest models. A user profile provides a task-centric approach to information gathering and dissemination. A profile allows structured management of user tasks and related information interests for each task. The objective is to build channels of information for each task that contains information relevant to a task's information interests. The structure of a user profile comprises a *task profile* and an associated *information interest profile*.

3.6.1. Task Model

A Task Model is a data structure to store task descriptions for each task. The architecture does not specify the form of task descriptions – they can be plain text descriptions or follow a specific or any other proprietary structure. Figure 1 shows the Task Model associated with relevant interests. A task description can be obtained by various methods which can be both manual and automated. The task description forms the input for determination of information interests described in the next section.

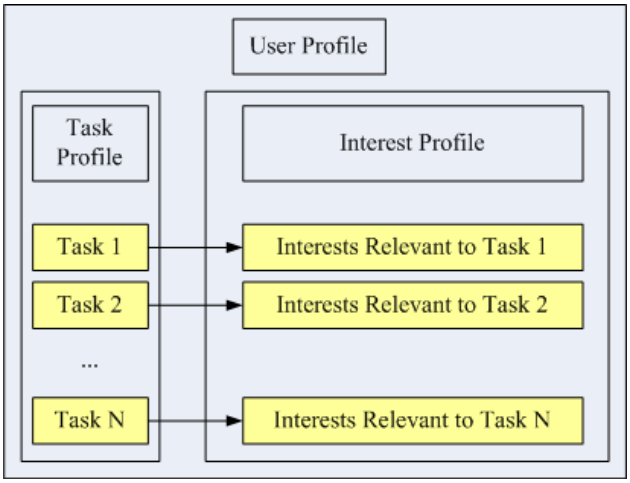


Figure 9: User Profile structure with its components, Task Model and associated Interest Model.

3.6.2. Information Interest Model

An Information Interest Model (Figure 10) is a data structure associated with each task in the Task Model and stores the information interests that are relevant to the task. The Interest Model is a multilevel nested structure that allows for storing interests and their relationships. The top level interests can be main interests and sublevel interests can be related or secondary interests. The architecture does not specify the interest representation form – they can be represented as plain text interests or in any other suitable structure. It also does not specify the actual relationships between interests. The forms of interest representation and relationships are left to the implementing application.

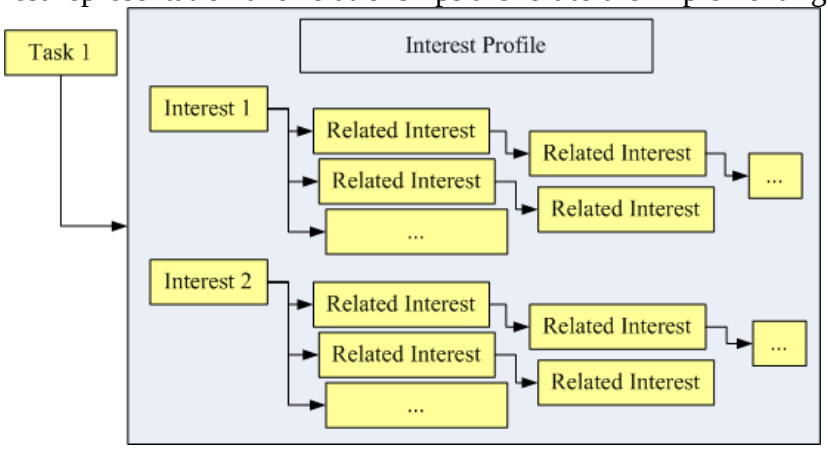


Figure 10: Interest Model as a multilevel interest storage data structure.

3.7. Profile Management

Profile management supports the following main set of activities.

- Task addition, updates and deletion
- Interest addition, updates and deletion
- Interest determination

These activities direct the agent's information gathering activities. In the following section, we describe a methodology used for information gathering and dissemination

that is based on the generalized architecture. The information representation is in text format such as text documents.

3.7.1. Information Gathering and Dissemination

An intelligence analyst, described as a user, can be considered to have intelligence analysis tasks. These tasks may have a detailed description or be vague as not enough information may be available to formulate the task. In either case, the TID architecture provides for an evolving method to find the right information and interconnect users that share common interests. The main steps in an agent's information gathering activity are:

- Obtain task description.
- Determine interests.
- Join appropriate VIGs for interests.
- Receive information disseminated through the VIGs.
- Analyze and annotate information with relevance scores.
- Display information to user.
- Obtain relevance feedback from user.
- Revaluate interests.

3.7.2. Information Gathering

Information gathering is task-centric. Interests are based on the tasks. These interests can be obtained by direct manual methods or by automated interest determination methods. We describe one such automated method below.

A plain text task description is analyzed using simple Natural Language Processing techniques to extract key concepts/actions about the task. Each individual concept is parsed to obtain keywords. A set of keywords form an interest, where the size of the set is 1 or more keywords. The set can also comprise related keywords to form semantically tight interests. The granularity of an interest can be set by the user. For a coarse grained setting, all keywords of the set form an interest while for a fine grained filter, individual keywords form individual interests. The determination of an interest is independent of the interest constitution of a VIG. A VIG's interest is always composed of a single interest regardless of the granularity of the interest. The granularity of the interest composition is left to the user or the agent. After the initial interests for a task have been determined, they are linked to the task by following the structure of the Interest Model. For each interest, the agent joins a VIG.

VIG join operation requires the agent to interface with the TID information network via an Information Router. A router allows the agent to search the network for an existing VIG for a particular interest. In case it finds an existing one, it joins the VIG, else it creates a new one. The VIG architecture allows for robust operation in that the failure of a VIG is not likely to be caused by the failure of anyone of the member agents, including the one that first created a VIG. In general, a structured P2P network is likely to fail if more than half of peers fail. A VIG discovery or creation involves creating a new router that represents the agent in the VIG. By spawning the router, the agent can be made independent of the information dissemination process. Using the TID architecture's multi-agent approach, the spawned router can be placed on other specific computing resources to maintain privacy and security of the disseminating information.

By joining the VIGs, the agent receives all information that is routed to the VIGs. Text documents are a form of encoding information. The VIGs form the first level information filter. Instead of the agent analyzing large volumes of information that may be irrelevant to the user's tasks, the agent now needs to analyze a much smaller subset. This is achieved by the VIG network as the documents routed to a VIG are relevant to the VIGs' interest. By configuring the granularity of the interests, the agent can decide how much of the information filtering is performed in the network. Coarse grained interests produce fine grained information filters while fine grained interests produce coarse grained filters. Thus, documents received through the VIGs by the agents indicate that they passed through the first level filter. The agent analyzes a received document and annotates it with similarity and novelty scores for each task. The scoring is based on the interests for each task. The premise is the interests for a task can help the agent in determining the value of information for the task. High similarity and novelty scores indicate that the information is valuable for a task.

Scored information is displayed to the user. A user can set information filter settings on the display so that only the information that passes through those filters is displayed. The user can change the scores and provide feedback to the user to indicate the true value of the information for a task. This feedback is usable in the future interest determination.

3.7.3. Interest Learning

Documents received through VIGs are annotated with scores indicating the value to each task. The documents are periodically analyzed to enhance the interests for a task. The premise is that the initial interests for a task may have been inadequate or incomplete for getting all information relevant to the task. By learning new interests, new VIGs may be joined, thus enlarging the scope of information gathering for a task.

Our current method for learning new interests is to obtain words that co-occur with the initial interests. This is done by performing a Latent Semantic Analysis using Singular Value Decomposition method. We investigated an advanced mechanism described in Appendix A.

3.7.4. Information Dissemination

Information Dissemination in TID generalized architecture is independent of whether there are users who may be interested in particular information. The disseminating entity can publish the information to the TID Information Network by interfacing to an information router. The router decodes the information into constituent features. The router maps the features to interests and cooperatively routes the information to all the VIGs that have the interests that were mapped from the features.

The routing infrastructure connecting the VIGs is based on a structured P2P network. The composition of the network is different from that of conventional server-based systems in that peers need very minimal setup to be started. Once bootstrapped, the routers join various multicast networks in the P2P network. The multicast network addresses are based on routers interests. This scheme enables routers with the same interests to join the same multicast network – thus forming the constituent network of the VIG. The flexibility of the peer software allows for routers to join and leave these networks at will. Message routing is based on the interests being resolved to the multicast address – i.e. to a particular interest – to the VIG and to each router in the VIG.

3.7.5. Information Processing

A main bottleneck to development of highly effective information processing is the large heterogeneity in the data. Large volumes of digital information on almost any topic are freely available from sources accessible via the Internet. These include online editions of print newspapers and journals, independent articles such as the blogs (short for web logs) and wikis and topic web portals. Such digital data sources offer information of varying quality and accuracy. Not all sources have scrupulous editorial services to assess the validity of information. In other cases, such editorial services may have been avoided as they require information about source qualities such as reliability and accuracy which requires additional inputs that are prohibitively expensive in terms of labor and time. Online data search engines such as Yahoo and Google provide scalable retrieval services but they lack crucial information about the information pedigree. Hence, it is often observed that data obtained through search results of web search engines is often fairly exhaustive (and sometimes exhausting to use) but not informative, novel and reliable enough to facilitate crucial decisions.

QLI's approach was to develop a language framework that identifies a section of the data space. If the data space is connected, e.g. domain – sub domain relations, topic-sub topic relations, the framework can be extended to crossover across the partitions. Such a framework allows for heterogeneous data to be automatically organized along human recognizable boundaries. A side effect of our approach is the identification of terminology that can be leveraged for knowledge discovery – such as to find related information and develop source information identities.

The framework is bootstrapped by a small number of training documents that clearly identify a particular data space – such as a topic. These documents are analyzed to determine term salience values. The salience scores are used to develop informative signatures of the document. The aggregate of topic training-document signatures form a topic signature. This can be replicated across other spaces – such as other topics or sub topics. Furthermore, the signature can be aggregated horizontally or vertically across the data space to obtain various levels of granularity. The framework is adaptive to allow the implementing system to evolve and match changes in informative signals or in user interests.

4. Implementation

QLI performed the research and development of the various aspects of TID in three distinct phases – proof of concept, demonstrator and prototype. The first two phases focused on developing the core architecture. The third and final phase focused on leveraging the core capabilities to meet the certain customer requirements.

4.1. Phase I – Proof of Concept

In the first phase, we demonstrated the proof of concept information network infrastructure. The Proof of Concept implementation of the TID architecture dealt with numerical information to demonstrate the main components of the architecture. An agent expresses its interest as an integer number. The features of the number are its unique prime factors. A VIG is associated with each prime number. The metadata about a prime number is the number itself. A router's address is derived from its associated prime number. An information message containing an integer number is sent to those VIGs that correspond to the number's prime factors. An agent provides additional filters that can be set by the user. The filter categories none, low, medium and high, correspond to the number of features (prime factors) that the received information must have. The first layer routers are developed using FreePastry[1]. FreePastry provides an API for a structured P2P overlay network. Information Routing and address resolution is based upon Distributed Hash Table. Performance evaluation is described in Appendix B.

4.2. Phase II Demonstrator

We implemented the main components of the TID architecture to demonstrate its key features. TID interface API specifications are described in Appendix C. RSS feeds were used to obtain news documents that were then stored in a database. A dedicated router was constructed to read documents from the database and publish them in the network. Each User Agent is provided with document indexing, storage, and text analysis modules.

The display browser (Figure 11) allows the user to input tasks and interests. The tasks form the root of a tree and its interests form the other nodes and leaves. Interests on a particular subtree indicate that they were learned for a particular interest identified by the root of the subtree.

The received documents are displayed along with relevant metadata about source, time of publication, and title. The information display GUI allows the user to set the information filters. We implemented a filter with four settings – high, medium, low, and none – each corresponding to different levels of document value. The document value is computed based on the weighted average of the number of VIGs providing a document to the agent, where the VIGs belong to the same task and the document similarity to the

[1] http://freepastry.rice.edu/

interests. The filter allows for viewing documents by task and interest that match the set filter.

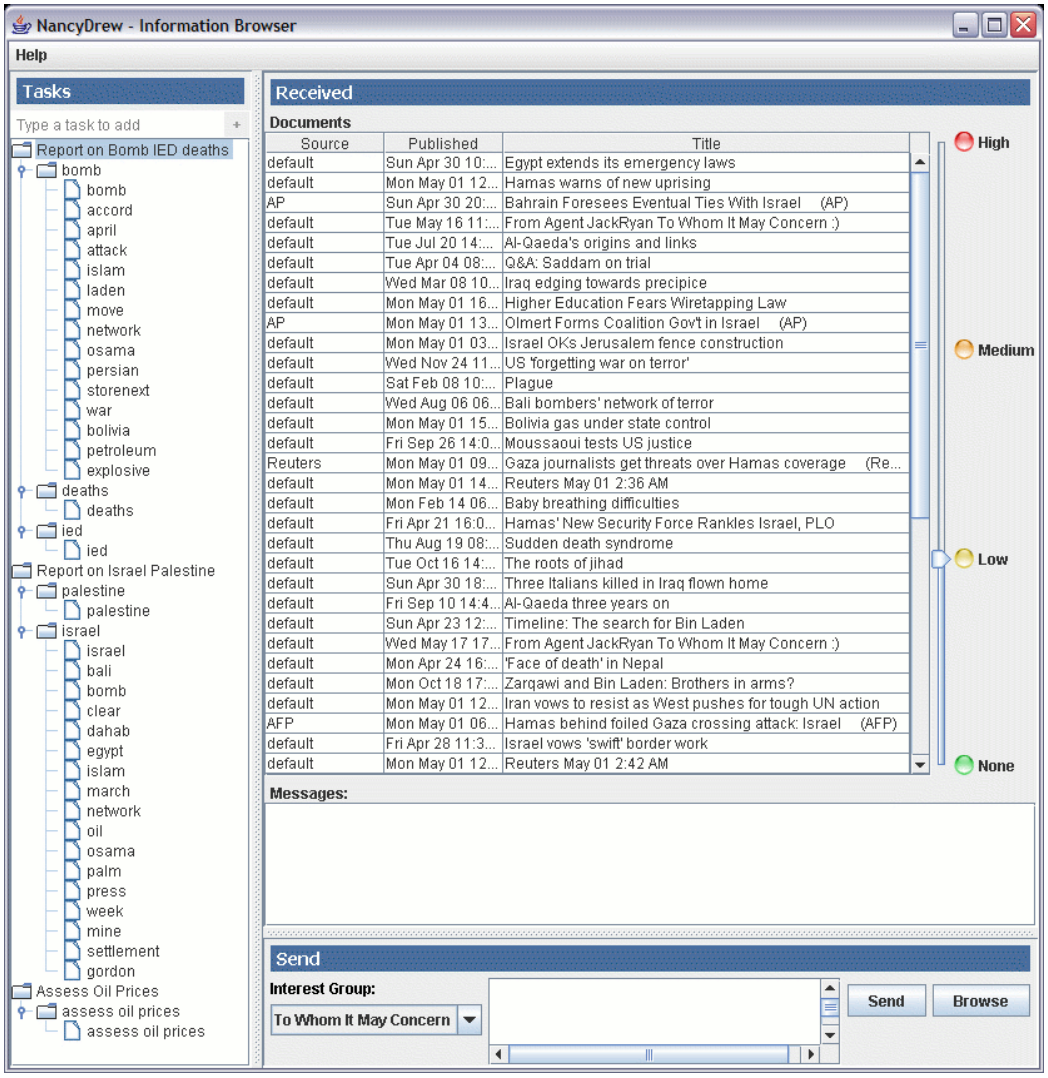


Figure 11: Information Browser displaying documents received by a User Agent. Tasks are displayed on the left along with its interests. The documents are displayed on the right, based on appropriate filter settings. The Send panel in the bottom is used to send a collaboration message either to a particular VIG or to anyone who may require the information. The later operation is done by a information router as part of the information dissemination process.

The GUI also allows for collaboration among members of a VIG. The premise is that users can share information with other members of a VIG once they've identified a need for information about a particular interest by joining the VIG. The implementation allows for anonymous collaboration by letting the network handle the dissemination of collaboration messages. The messages could be sent to members of a particular VIG or by routing based on the content of the document.

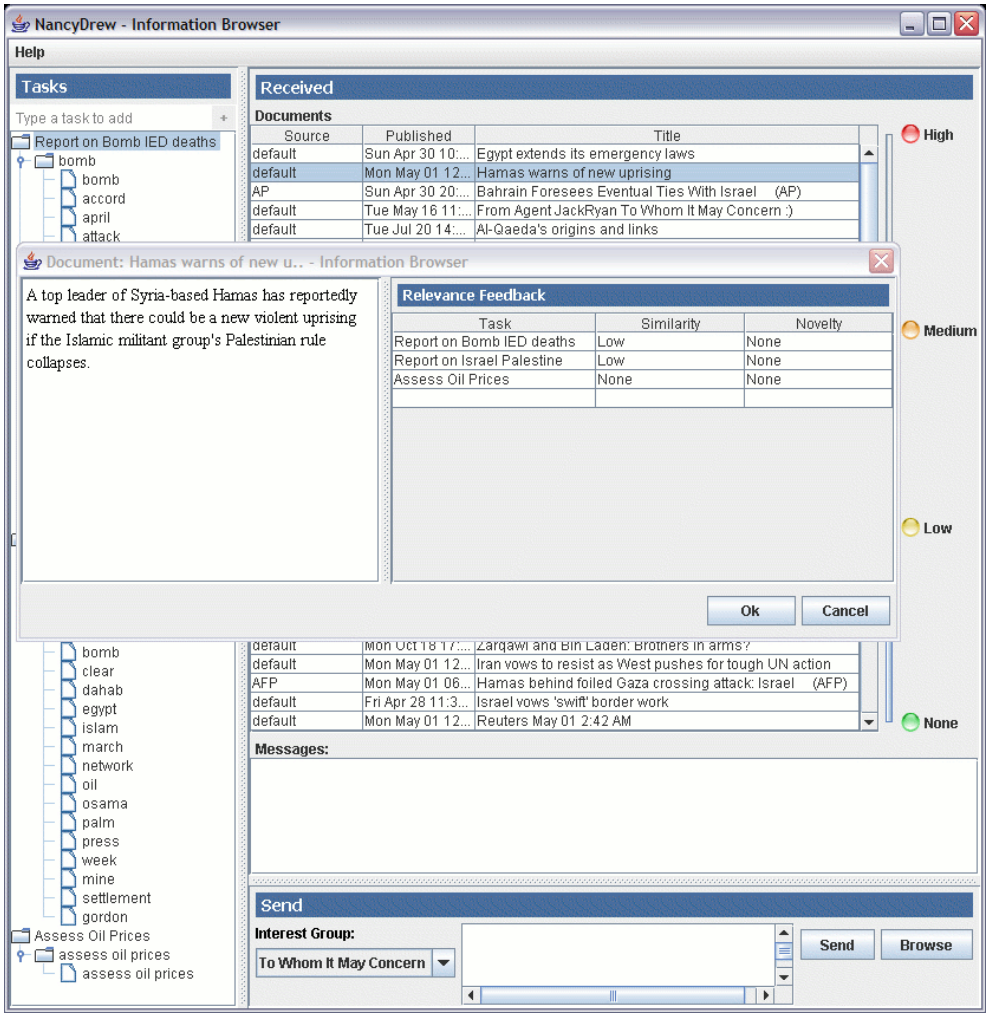


Figure 12: Document Relevance Feedback GUI.

A document's information value is viewed by opening the document in the browser. Relevance feedback can be provided to the agent by the user (Figure 12).

5. Phase III Prototype:

5.1. Motivation

U.S. Northern Command (USNORTHCOM) is tasked with providing command and control of DOD homeland defense efforts and coordination of defense support of civil authorities. It is responsible for the Continental US, Canada, and Mexico, and also has global responsibility with respect to epidemic or pandemic situations. With respect to the challenge of identifying a biological incident, USNORTHCOM has documented their 'need' as shown in Figure 13.

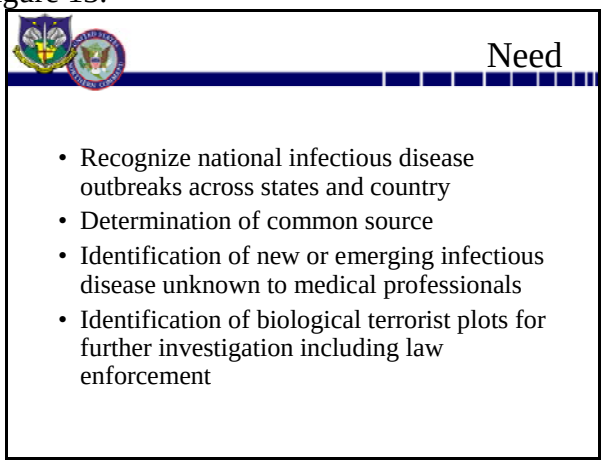


Figure 13: USNORTHCOM needs for 'Identifying a Biological Incident'

Furthermore, USNORTHCOM have identified that effective responses to a biological incident are more readily implemented and are more successful if the event is identified as early as possible. In Figure 14, the components of discovery and their impact are outlined.

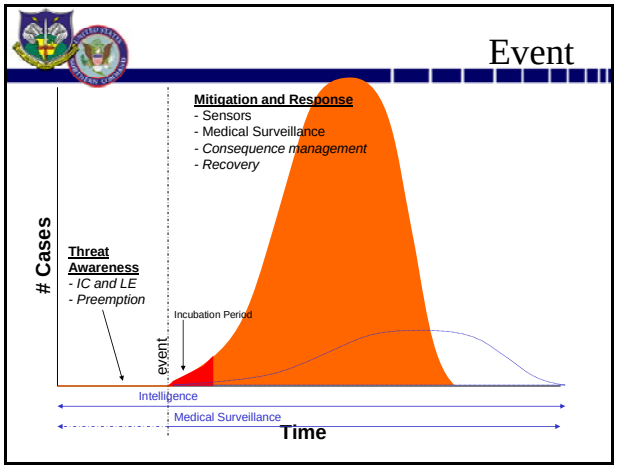


Figure 14: Early Warning & Impact on a Biological Incident

Based on the requirement for as early a detection as is possible, it is clear that an early warning system (EWS) is required to assess the identification of, and status of, an outbreak to enable effective planning and coordination of the response. Such a system would be informed by sources as follows:

- Information indicating a incipient outbreak or cluster of disease cases is likely to first be available from fragmented and incomplete sources - local news agencies, agricultural groups, scientists involved with infectious diseases, etc.
- Later reports will be from more authoritative sources and include corroborative data – These sources may include larger news agencies and disease specialists. Frequently these reports are collated and available through tracking sites such as ProMed. Specific sites for monitoring early indicators of a disease episode – examples include ESSENCE, GEIS, RODS.
- Finally, authoritative reports from sources such as the CDC or WHO will be issued after substantial evidence has been collected and vetted.

Because it is important that initial planning is accomplished even ahead of confirmed epidemic reports from authoritative sources, USNORTHCOM will need to consider and assess reports that are highly sporadic and variable in quality and reliability, and must provide a "best assessment" for leaders in the chain of command, along with estimates about the reliability of the assessment, based on the quality and reliability of the evidence documents. To address these limitations, QLI proposes a novel solution to EWS for PI. The proposed EWS would provide a capability consistent with the concerns documented by USNORTHCOM as obstacles to effective EWS or in their terminology the 'big rocks'.

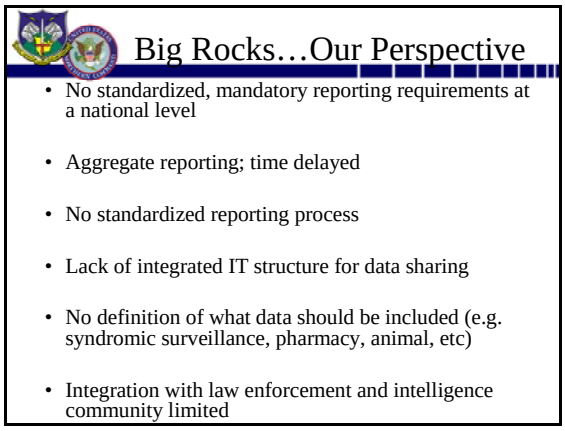


Figure 15: USNORTHCOM documented "Big Rocks'

In their consideration of possible solutions to the stated problem, USNORTHCOM identified several potential sources of assimilated or even raw data; some of these are shown in Figure 15.

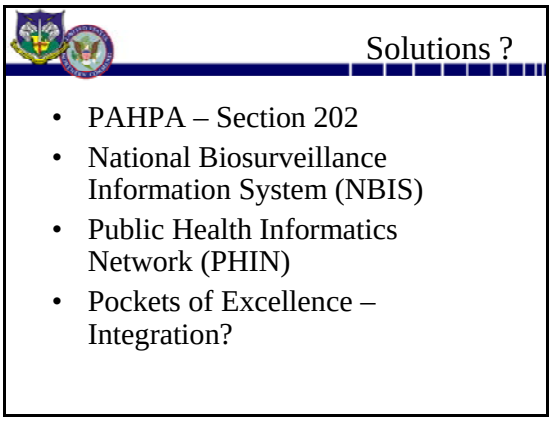


Figure 16: USNORTHCOM documented solution providers

All of those sources identified in Figure 16 can be linked to the EWS that is being proposed. However, the utilization of these sources linked in a single enterprise to an even larger set of resources will provide an enhanced capability and improve the chances for successful early detection. The drawback with a larger set of sources is the volume and velocity of data that needs to be handled and the extraction from that 'sea of data' the one or two key pieces of information that can generate triggers for action on the part of the user. It is the combination of the two elements described into one EWS that makes the proposed system so compelling. The fact that the basis for the system already exists in current Quantum Leap developed technology will shorten the development time, saving costs and quickly moving from 'nice to have idea' to a usable EWS.

To address these issues, QLI is developing the architecture for Targeted Information Management (TIM) in open data environments to attain domain specific situational awareness. Specifically, with respect to the problem as stated above, QLI is developing TIM architecture for attaining situational awareness in the domain of infectious diseases. TIM provides an early warning system for responders and coordinators of a response to a biological incident that is capable of assessing the status of infectious diseases based on a large source of (relevant) data.

Because it is important that initial planning is accomplished even ahead of confirmed epidemic reports from authoritative sources, response coordinators will need to consider and assess reports that are highly variable in quality and reliability, and must provide a "best assessment" for leaders in the chain of command, along with estimates about the reliability of the assessment, based on the quality and reliability of the evidence documents. Development of TIM will provide a basis for 'triaging' the data with a target of turning data into actionable information.

5.2. Overview of TIM

The goal for TIM will be to 'access' data and 'present' the data, either singularly or in combination, as information with a representation of the 'quality' of the data as a decision support capability. TIM is constructed to provide representation and maintenance of data pedigree and quality of the data from access or retrieval to presentation or supply as information. TIM infers data provenance and redundancy by comparing the accessed documents so that decision-makers are able to distinguish between multiple duplicates of a single account versus multiple independent accounts –

this can be considered as 'data lineage or pedigree'. In addition to tracking data lineage, TIM algorithms can infer 'qualities about the data source based on corroboration by reputed and reliable sources and user feedback (when available).

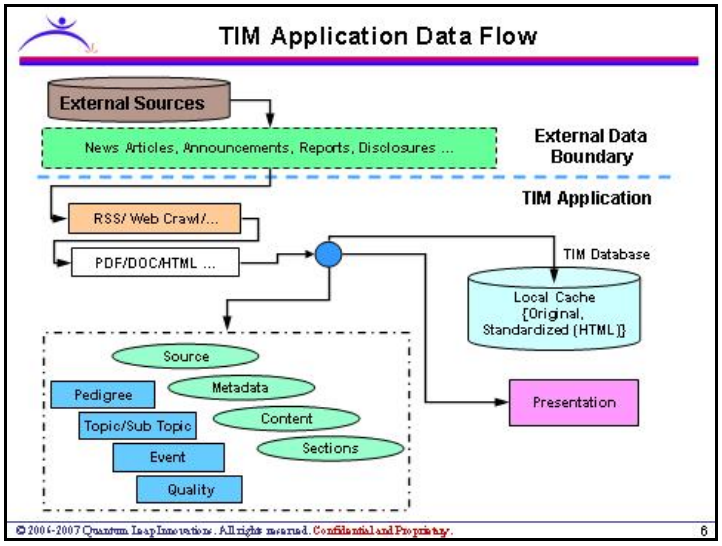


Figure 17: TIM Application Data Flow

The initial focusing problem for TIM development has been to access data concerning infectious disease(s). Specifically, in the initial phase of TIM development, the focus has been on data concerning current incidences of avian flu, human infections resulting from infected birds, and, the resultant possibility of a pandemic flu outbreak. This focus is consistent with the motivating problem as outlined for USNORTHCOM in the first section of this document.

The information pertaining to infectious disease often results from the work of highly specialized sources of data. Moreover, it is likely to be reviewed by a wide array of personnel, with varying training, information needs, and roles. TIM triages reports into appropriate topics that can be examined, browsed, or subscribed to, by different (or appropriate) personnel. TIM data flow (Figure 17) is described below:

- In the first step (Figure 17 – top left quadrant) data sources are identified and connections are established. Periodically or as new data is available, the data source connections are invoked and documents obtained. Syndication technologies such as RSS and Atom provide for rapid and immediate data gathering capabilities. The documents are internalized to facilitate TIM processes. The main steps of internalization are to convert documents on standard presentation and viewing formats – such as HTML and to stamp the document with TIM system wide GUID (Globally Unique Identifier).
- In the second step (Figure 17 – bottom left quadrant – green ovals), a cached document is analyzed to extract available metadata and content. The document content is additionally parsed to obtain salient sub-documents. These data artifacts of a document are cached for further processing.
- In the third step (Figure 17 – bottom left quadrant – blue boxes), a stored document and its sub-documents are analyzed to identify & describe their salient aspects. The salient aspects allow for development of a wide ranging set of data

characteristics. In addition, the document's pedigree is computed. The characteristics along with the pedigree provide for presenting information for examination and browsing - such as for viewing all information per event, archived information, by locations and by data source. They also allow for further machine analysis such as identifying duplication and/or corroboration of specific information and data sharing across applications.

- In the final step (Figure 17 – right half) the data is presented via the TIM user-interface (UI). Access to the UI can be via a web-browser or directly from the network on which TIM is operating.

5.3. TIM Architecture -- Situational Awareness

The TIM architecture that will support the gaining of situational awareness comprises the following components:

Data Component

- Marshall data from sources
- Identify, develop and update data characteristics
- Triage, track and archive information with respect to data characteristics

Pedigree Component – with the following functions:

- Bootstrap known and identify new data sources
- Track and record data pedigree
- Compute/update information quality from available evidence
- Source reliability, data duplication, authority citation, data duplication, information corroboration

Presentation Component

- Organize and display Information
- Interactive filters and queries
- Parametric subscription for real time updates

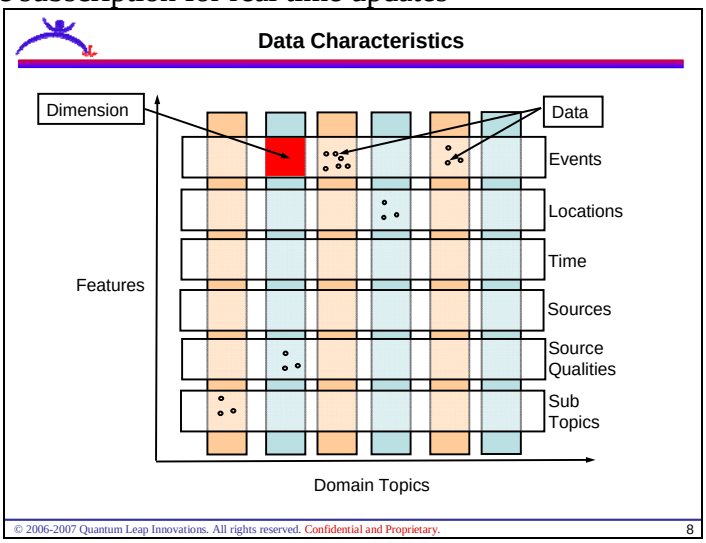


Figure 18: TIM Dimensions based on data Characteristics/Features

The combination of the TIM components provide capabilities to identify and develop data characteristics that provide for data to be viewed, visualized and made available for

further analysis. The principle data characteristics are domain relevant topics. These primary characteristics are supported by domain wide set of features. The features include but are not restricted to events, locations, time, sources, source qualities, sub-topics. The combination of the two provide for categorizing information along various dimensions. For example (Figure 18), dimensions can allow for categorizing data about an *event* from all *sources*, *latest* data from the *most reliable sources* and so on.

Data categorization along these dimensions enables the basis for information tracking and quality computation. Often, a single information source is duplicated and re-reported multiple times. In other cases, a single incident is reported with different interpretations with varying accounts of the facts. In more sinister cases, information may also be misrepresented. In addition, information tends to be available in bits and spurts and irregular intervals. The TIM pedigree component builds upon the data categorization to track data that are about single and a set of related dimensions. In addition, the data categorization provides for identifying data pedigree based on related content. For example, if a reliable source provides data about an incident that corroborates data provided earlier in time by another source, TIM can establish likelihood of *information pedigree* based on the relatedness of the data. In general, TIM utilizes available data references and computes implicit linkages to track and establish data pedigree.

5.4. TIM System View

Data categorization along with data pedigree provides for data tracking capabilities for specific sets of interests and parameters developing a rich set of descriptive features about various past, current and emerging incidents (Figure 18). The parameters can be used to create subscriptions to receive constant updates about information of interest. The rich set of descriptors computed by the TIM components provide for intuitive data presentation and visualization. Figure 19 depicts a generalized view of the TIM data presentation layer. The data descriptors provide data to be presented in different views. A set of filters can be developed for data to be viewed along any of the dimensions.

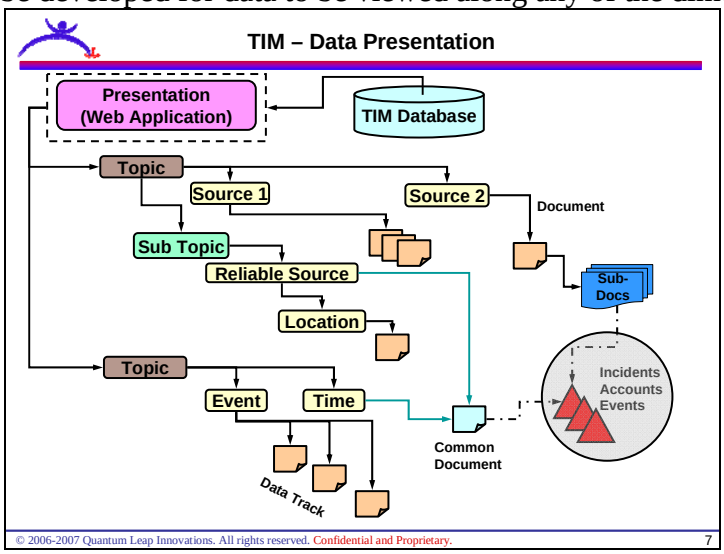


Figure 19: Data Presentation

5.5. TIM Prototype Solution Development & Transition

The main features of the web application are:

- Organize information by topics and features
- Interactive filters
- Present information pedigree and quality
- Sorting functions per feature

The web application (Figure 20) monitors a wide range of data sources on the open internet and gathers vast amounts of data periodically since it is the goal of this application to triage and present PI incident/event related information in real time.

The application has three main tiers of data sources with respect the PI domain. Tier 1 comprises known or reliable sources – CDC, WHO, etc. Tier 2 comprises known news sources such as AP, NY Times, BBC, etc. The third tier comprises all the other sources that may not be well known or reliable. A set of topics are seeded into the application by PI experts. These topics provide the initial set of interesting categories for the users. The TIM system develops identifying signatures for these topics to filter documents (data) from above mentioned data sources. Information not relevant to the domain is filtered out. The selected document is analyzed to extract features (source, location, time references) and is categorized into appropriate topics. The document is also analyzed to extract other salient features such as number of cases, types of strains, etc.

Document pedigree is computed by identifying explicit references and computing content based similarities. The features provide an organization of information based on descriptions of the events and incidents. Document pedigree is used to accumulate and build corroborative or refutable evidence from available data such as data source reliability and separate accounts of the events or incidents. Unknown-source reliability is evolved based on evidence from information that has been vetted or refuted by reliable sources. A data signature is computed that determines the authoritativeness of the sources. Past and vetted data is used to improve the topics and their signatures. All evidence and data used to arrive at a particular feature, data description or conclusion is available for examination. An interactive web-site provides for intuitive browsing capabilities. The interactive filters provide for information of interest to be displayed.

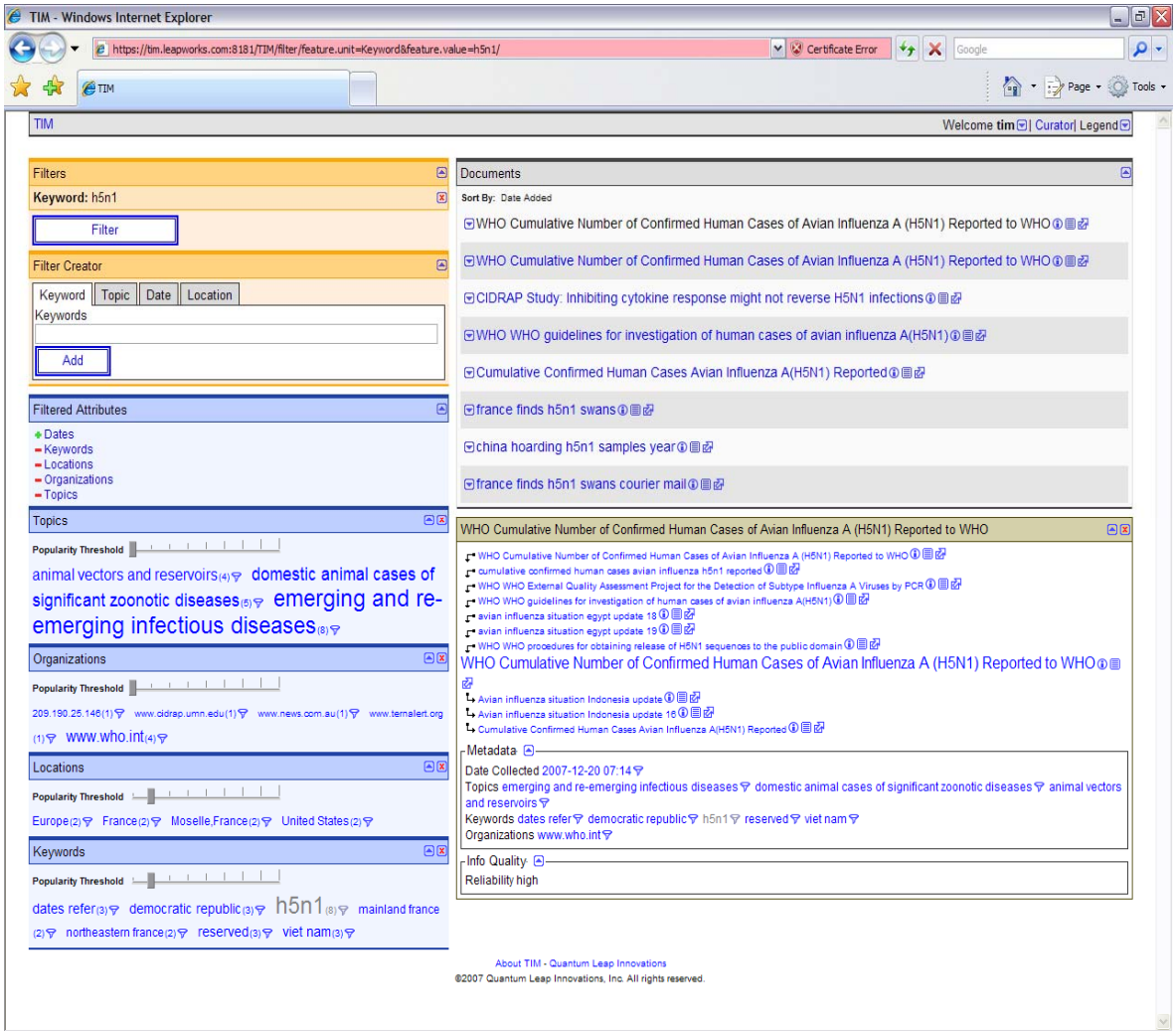


Figure 20: TIM Web Application – Home page snapshot

6. Conclusions

This report provides a description of the TID system architecture. The proof of concept tests show that the two layered approach where each layer provides different granularities of information filtering and routing is a scalable solution for disseminating high volumes of information to a large user base. The ability to effectively disseminate information to analysts while maintaining system efficiency provides for vital assistance to an intelligence analyst. We demonstrated the core capabilities of the TID architecture and implemented the main components through the demonstrator. The implementation provided us with valuable insight about scalability and effectiveness of the architecture. The profile management methods gave us direction to implement effective interest learning algorithms that harnessed the distributed nature of the architecture. Finally, we developed a prototype solution for the USNORTHCOM to meet their situational awareness requirements.

Appendix A: Text Mining for Interest Profile Management and Learning

We consider Information gathering as a task-centric operation where the user interests are based on the tasks. These interests can be obtained by direct manual methods or by automated interest determination methods. The following sections describe two methods for extracting and learning interests.

Interest Learning

A plain text task description is analyzed using simple Natural Language Processing techniques to extract key concepts/actions about the task. Each individual concept is parsed to obtain keywords. A set of keywords form an interest, where the size of the set is one or more keywords. The set can also comprise related keywords to form semantically tight interests. The granularity of an interest can be set by the user. For a coarse grained setting, all keywords of the set form an interest while for a fine grained setting, individual keywords form individual interests. The granularity of the interest composition is left to the user or an agent.

An agent gathers information for each of the interests. In our methodology, the agents join Virtual Interest Groups or VIGs for each of their interests. A network of such VIGs enables the agent in gathering information relevant to the interests. The VIGs form the first level information filter. Instead of the agent analyzing large volumes of information that may be irrelevant to the user's tasks, the agent now needs to analyze a much smaller subset. This is achieved by the VIG network as the documents routed to a VIG are relevant to the VIGs' interest. By configuring the granularity of the interests, the agent can decide how much of the information filtering is performed in the network. Coarse grained interests produce fine grained information filters while fine grained interests produce coarse grained filters. Thus, documents received through the VIGs by the agents indicate that they passed through the first level filter. The agent analyzes a document and annotates it with *similarity* and *novelty* scores for each task. The scoring is based on the interests for each task. The premise is the interests for a task can help the agent in determining the value of information for the task. High similarity and novelty scores indicate that the information is valuable for a task.

Scored information is displayed to the user. A user can set information filter settings on the display so that only the information that passes through those filters is displayed. The user can change the scores and provide feedback to the user to indicate the true value of the information for a task. This feedback is usable in the future interest determination.

Documents received through VIGs are annotated with scores indicating the value to each task. The documents are periodically analyzed to enhance the interests for a task. The premise is that the initial interests for a task may have been inadequate or incomplete for getting all information relevant to the task. By learning new interests, new VIGs may be joined, thus enlarging the scope of information gathering for a task. New interests are learned by computing words that co-occur with the initial interests. This is done by performing a Latent Semantic Analysis using Singular Value Decomposition method.

Text Classification

The problem of assigning documents to a predefined set of interest groups appeared to be a variation on text classification - with one exception, text classification traditionally placed a document into a single category. For simplicity's sake, the initial experiments follow traditional document classification and limit each document to a single interest group. However, we discuss plans for expanding text classification to multiple interest groups.

To process the documents, the documents are first converted into bag-of-words statistics: a count of how many times each word appears in each document. BOW, or "Bag of Words", simply records the instance counts; it makes no attempt at capturing the words' context or any syntax. Full BOW statistics for an entire corpus quickly become unwieldy. Several researchers have shown that clustering words and then calculating cluster-counts can effectively reduce the number of features for each document without sacrificing accuracy [1] [2]. In some cases, cluster-counts outperform raw BOW, especially with sparse data.

While several researchers have shown the benefits of using more-complicated algorithms (e.g. Information Bottleneck or Distributional Clustering for word clustering, or Information-Theoretic co-clustering for unsupervised clustering [1] [2] [3]), we chose to start with simple algorithms: k-means to cluster the words, and Naive Bayes Multinomial for document classification. The simple algorithms allowed for leveraging established machine-learning libraries and a more-rapid look at interesting questions. It also provides a baseline for evaluating future experiments.

Additionally, we plan to expand the predefined interest groups using unsupervised document clustering. Since both the supervised and unsupervised approaches use the same underlying data, we posit to find a relationship between the clustered documents and the predefined interest groups. If the number of clusters is an order of magnitude larger than the number of interest groups, presumably some of the clusters will fall completely within a single interest group, while others may fall along the border, spanning multiple interest groups. Clusters could be considered VIGs in their own right, and an agent could be assigned to a VIG based on the percentage of documents they already receive from their interest groups.

Alternately, by clustering a known set of documents, we could calculate the probability that a member of a given cluster belongs to an interest group. We could then combine evidence from the supervised classification with evidence from the unsupervised clustering to produce our final probability distribution.

The project tasks were

1. Build the classifier.
2. Build the clusterer.
3. Combine them.

Supervised Clustering

Several studies have shown that documents can be classified with a high degree of accuracy. Our objective was to investigate different methods for routing documents to an agent who is interested in multiple interest groups. Our first approach involved classifying each document into a single interest group. However, Naive Bayes

Multinomial produces a probability distribution for a document over all interest groups. Ambiguous documents may have relatively equal probabilities across several interest groups. Simply assigning the document to a single group drops potentially useful information.

We calculated the probability that the document is relevant to an agent (the sum of the probabilities over all the agent's interest groups) and comparing that with the probability that the document is irrelevant (the sum over all other interest groups). Since $relevant = 1.0 - irrelevant$, we only need to calculate the relevant probability. If the relevant probability is greater than 50%, the document is sent to the agent. Otherwise, it is not. Finally, we could adjust the 50% threshold to fine tune recall vs. precision.

In this simulation, we assume that there is a service in the system of agents that knows the agent's interests. The simulation comprised a number of mock agents, with each agent interested in two interest groups. A known corpus is processed through the classifier and routed to agents based on the following methods that determine whether the document could be sent to an agent.

1. Sum: Over all interests of an agent.
2. Crisp: Over each interest.

If the document was correctly sent to the agent, it counted as a positive hit. If the document was incorrectly sent to the agent, the document counted as a false positive. Finally, if the document was incorrectly not sent to the agent, it is a false negative which led to the calculation of recall, precision and f-score for each agent.

1. Recall is the percentage of relevant documents correctly sent to the agent.
2. Precision is the percentage of documents sent to the agent that were actually relevant.
3. F-Score is the harmonic average of the two, and can be used as a general performance measurement.

$$recall = \frac{positive}{(positive + falsenegative)}$$

$$precision = \frac{positive}{(positive + falsepositive)}$$

$$f - score = \frac{(2 \times recall \times precision)}{(recall + precision)}$$

We then compared the recall, precision and f-score of the crisp-classification assignments with the recall, precision and f-score of the probability-summing assignments.

Experiments:

We used the 20-newsgroup corpus for all experiments [4]. This corpus contains 11,078 training documents and 7,388 test documents, taken from 20-different Usenet newsgroups. The corpus maintainers have removed duplicate posts and stripped all newsgroup-identifying header information. For the purposes of these experiments, we considered each newsgroup to be its own interest group; however, many of the newsgroups are closely related. Several groups discuss different aspects of related topics: particularly computers, sports, science, religion and politics. To create the BOW

statistics, the words were parsed using the Lucene StandardAnalyser (with default stop words) [5], then low frequency words (any word that appeared less than four times in the corpus) were filtered out. This left 29,987 relevant words in the training set.

Creating Cluster Maps

To cluster the words, we counted the number of times each word occurred in each interest group. This resulted in 29,987 word-instances, each with 20 features. These instances were then clustered using k-means.

To determine the number of clusters, we tried running the data through WEKA's EM Clustering algorithm [6]. This implementation ran cross validation to determine the number of clusters. We hoped this would give a good first guess at the best cluster size. Unfortunately, it returned only 9 clusters—which seemed suspiciously small. So, we decided to test several different cluster sizes: 10, 100, 1,000, 2,000, 5,000 and 10,000 clusters. We then produced the corresponding cluster maps and saved the word-to-cluster mappings to disk. Again, we created the cluster maps using only data from the training set. We calculated cluster count statistics for both the training and test sets—producing a data file for each cluster size over each corpus. The different cluster sizes were tested by running 10-fold cross validation experiments using WEKA Experimenter. For each data file, experimenter ran 10 random, 10-fold cross-validations (for a total of 100 runs). The averages are shown below:

Trials:	Pre.	Rec.	F-Score	Build Time	Test Tfime
Training All Relevant Words	89.7551	91.3617	90.4683	0.3750	0.2519
Training 10 Clusters	18.0116	30.8511	22.6834	0.0168	0.0261
Training 100 Clusters	88.6970	88.8298	88.6555	0.0712	0.0474
Training 1,000 Clusters	93.1846	96.7234	94.8722	0.1021	0.0773
Training 2,000 Clusters	94.1853	96.8298	95.4482	0.1160	0.0822
Training 5,000 Clusters	94.5720	97.1277	95.8024	0.1467	0.0988
Training 10,000 Clusters	94.2525	96.4894	95.3142	0.1920	0.1280
Test All Relevant Words	80.5205	89.4567	84.5991	0.2544	0.1376
Test 10 Clusters	15.7110	24.4940	18.9842	0.0103	0.0134
Test 100 Clusters	65.2386	67.9657	66.2840	0.0478	0.0309
Test 1,000 Clusters	75.4796	79.1028	76.9928	0.0687	0.0519
Test 2,000 Clusters	78.2494	82.6129	80.1288	0.0772	0.0592
Test 5,000 Clusters	80.3845	85.6028	82.6937	0.0993	0.0689
Test 10,000 Clusters	80.0323	89.3327	84.2580	0.1408	0.0864

On the training set, using more than 1,000 clusters produced better results than using all the relevant words. The scores peak at 5,000 clusters and begin to drop again. This is not surprising, since the clusters were created to capture the relationship between words and interest groups specifically over these documents.

On the test set, all scores dropped. Even the all-relevant-word scores came out significantly lower. We found this especially interesting, since the list of relative words

came from the test corpus itself (all non-stop words in the test set that occur at least 4 times). Most of this drop probably came from the change in corpus size. The test data is 1/3 smaller than the training data. Since we used 10-fold cross-validation for all runs, the test set's classifiers were built using less information.

In the test set, larger numbers of clusters always outperformed smaller clusters, and using all relative words produced the best results. These experiments give a good indication on how word-clusters will perform over new documents.

While we could design additional experiments to try and clear up some of the outstanding questions, these results seem sufficient for picking a reasonable cluster size. When you compare classifier build times, test times and F-score, the 5,000-cluster appears to sit at the sweet-spot, balancing efficiency and accuracy.

Testing Crisp vs. Sum Document Classification

We created a mock agent for every unique interest group pair (190 total mock-agents) and decided to look at all pairs, rather than just a subset of pairs, since different pairs will likely respond differently. For example, when classifying the documents into crisp groups, closely related pairs are likely to have a number of documents misclassified from one member to the other. Since the document still matches the pair, it counts as a positive match even though the original classification was incorrect. Pairs that differ greatly are more likely to have misclassified documents fall outside the pair.

We used the test set's 5,000-cluster data. Again, we ran 10 trials and each trial used 10-fold cross validation. During each trial, I tracked the positive, false positive and false negative hits for each pair. The averages over all pairs are listed below:

Type	Recall	Precision	F-Score
Crisp	95.53	95.58	95.53
Sum	95.5	95.73	95.58
Sum - Crisp	-0.03	0.14	0.06

Summing tends to do worse on recall, but improves precision and f-score slightly. While the F-Score shows a 0.06% overall improvement when summing over the distributions, the difference between the two approaches is essentially negligible. No score varied by more than 0.5%. Interestingly, the paired scores came out much higher than the single interest-group scores. This makes some sense; the chance of randomly assigning a document to the correct group has doubled (jumping from 5% to 10%). We expected the highest scores to be in closely-related pairs, but just glancing over the data, that doesn't seem to be the case. Here are the top 10 pairs for both recall and precision:

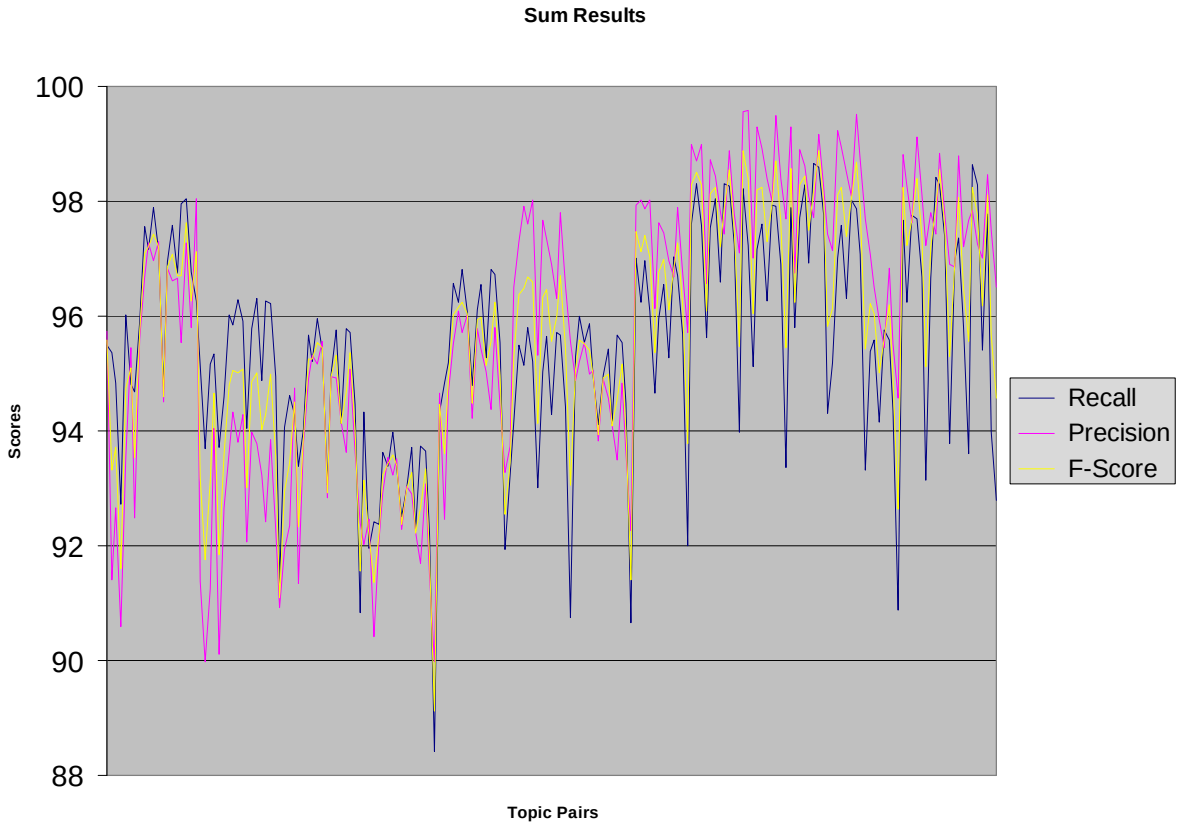
Recall:

rec.sport.hockey--talk.politics.guns
talk.politics.guns--talk.politics.mideast
rec.sport.hockey--talk.politics.mideast
sci.space--talk.politics.guns
rec.motorcycles--talk.politics.guns
rec.motorcycles--rec.sport.hockey
rec.sport.hockey--sci.space

sci.space--talk.politics.mideast
talk.politics.guns--talk.politics.misc
rec.motorcycles--talk.politics.mideast

Precision:
rec.sport.baseball--sci.crypt
rec.sport.baseball--rec.sport.hockey
sci.crypt--talk.politics.mideast
rec.sport.baseball--talk.politics.mideast
rec.sport.baseball--sci.med
rec.sport.hockey--sci.crypt
sci.crypt--sci.med
rec.sport.hockey--talk.politics.mideast
sci.med--talk.politics.mideast
rec.motorcycles--sci.crypt

Even a casual glance over this lists shows that the same interest groups keep appearing. This seems to imply that high-performing pairs are produced by combining high-performance interest groups. Indeed, rec.motorcycle, rec.sports.hockey and talk.politics.mideast are the top three performers for recall. For precision: rec.sport.baseball, rec.sport.hockey and talk.politics.mideast. On the other hand, talk.politics.guns is a mediocre performer—but its errors are clustered and correspond with higher performers. To get a better idea, look at the sample confusion matrix [Appendix D]; however, take these numbers with a grain of salt. The confusion matrix is based on a single run. Below is a graph showing all sum scores. Again, the crisp scores are virtually identical.



Conclusion

The difference between the crisp and sum document classification techniques are negligible. TID's current method of distributing data more-closely resembles the crisp classification approach. Data is placed into interest groups, and each agent pulls the data from all their interest groups. Summing over interest groups would require a fairly significant change in the system—given the results of these experiments, any change seems unwarranted.

Unfortunately, some of the results don't make much sense (e.g. why is rec.sport.hockey--talk.politics.guns the top performer for recall?). On the one hand, the confusion matrix is based on a single run. The experimental results are the average over 100 runs. Perhaps talk.politics.guns just performed poorly on this particular split. However, this inconsistency may indicate a bug in the trials or an error in math, and deserves further investigation.

Unsupervised

We proposed clustering the documents into a large number of clusters, roughly 10x the number of interest groups, to start with. Since the current corpus has 20 interest groups, we would use k-means to separate the documents into 200 clusters. As in the supervised case, we hoped to reduce the document features to a more manageable number. However, clustering the words based on the number of times they occur in an interest group seemed inappropriate. That would inject supervised knowledge into the unsupervised learner.

To keep a more-pure unsupervised environment, I proposed clustering words based on the number of times they co-occurred with a sentence. Words that frequently

appear together often have similar or complimentary meanings (e.g. “car” and “drive”). By clustering words based on co-occurrence over a large number of documents, we should capture the functional relationships between words in our corpus. Unfortunately, both the number of features and the number of words is very large. If we have N words, the co-occurrence data is an N x N table. With N close to 30,000, k-means runs unacceptably slow, even when producing only 100 clusters. Instead, we tried WEKA's farthest first traversal algorithm; a fast cluster approximation algorithm based on k-means. While faster, this algorithm still could not complete the 10,000 cluster sets in a reasonable amount of time. The size of this data may remain a problem as we move to more advanced algorithms. One option is to more-aggressively restrict the initial relevant words—or perhaps restrict the words across one axis (only tracking co-occurrences with a more-limited selection of keywords).

We created sentence cluster maps for 10, 100, 1,000, 2,000 and 5,000 clusters. We have not tried using these data-sets to cluster the documents; however, I ran them through WEKA’s NBM classifier to compare the results with the interest-group based clusters.

Sentence-Cluster Averages	Precision	Recall	Fscore
train 10	0.09742	0.127021	0.109627
train 100	0.277599	0.328085	0.299676
train 1,000	0.71405	0.737447	0.723613
train 2,000	0.791698	0.788936	0.788465
train 5,000	0.852468	0.872979	0.86138
test 10	0.100233	0.128286	0.11095
test 100	0.228631	0.255111	0.239448
test 1,000	0.603208	0.685847	0.63952
test 2,000	0.681214	0.772067	0.721056
test 5,000	0.733668	0.837369	0.779736

These scores are worse than the interest-group clusters; which is expected. The interest-group clusters captured the words’ relationship to the interest groups. Sentence co-occurrence clustering is completely unsupervised. Still, as an initial sanity check, the results appear reasonable. Next, we would like to cluster the documents and look at how the clusters relate to the interest groups.

Future Projects

There are a number of possible future projects that could come from this work.

- Finish the unsupervised approach and look at ways of combining the supervised classifiers and unsupervised clusterers.
- Expand the supervised classifier to classify documents into multiple interest groups. One option involves creating several parallel binary classifiers—one classifier for each interest group. Each classifier then identifies the document as either belonging to or not belonging to a specific group. Ideally this would be trained on a corpus with documents that belong to multiple categories.
- Explore alternate algorithms for clustering/classifying. Distributional Clustering [2] looks promising for the supervised word clustering. Information-Theoretic Co-Clustering [3] looks promising for the unsupervised case.
- Given a starting cluster map, we could use machine learning techniques to improve the cluster map (genetic algorithms, reinforcement learning, etc.).

References:

[1] N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In SIGIR 2000, 2000.

[2] L.D. Baker and A.K. McCallum. 1998. Distributional clustering of words for text classification.

[3] I. S. Dhillon, S. Mallela, and D. S. Modha, Information-theoretic co-clustering," in Proceedings of The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD-2003).

[4] 20 Newsgroup Data Set: http://people.csail.mit.edu/jrennie/20Newsgroups

[5] Apache Lucene: http://lucene.apache.org/java/docs/

[6] Ian H. Witten and Eibe Frank (2005) "Data Mining: Practical machine learning tools and techniques", 2nd Edition, Morgan Kaufmann, San Francisco, 2005.

Appendix B: Experiment Setup, Results and Performance Analysis

We conducted an initial set of experiments to evaluate information dissemination performance. The experiments were conducted on a 16 dual processor nodes on a Beowulf cluster. Each node had 2 GB Memory. The test setup was:

- 3 Interests per agent (3 integer numbers, randomly assigned).
- Number of agents 48 to 160. (3-10 per node).

Information dissemination volume – 10000 messages at the rate of 1 message per second
A single dedicated agent randomly generated numbers and injected them into the system. The numbers were resolved to overlay addresses and sent to "exact" matching routers. The system discarded messages which corresponded to non-existing routers. On average, each integer number resulted in 2.3 VIGs. That is, the average number of unique prime numbers per interest was 2.3. The minimum test configuration was 16 agents, 16 interests (not unique), 38 VIGs (unique) and 100 information messages while the largest test configuration was 160 agents, 480 interests (not unique), 1104 VIGs (unique) and 10000 information messages.

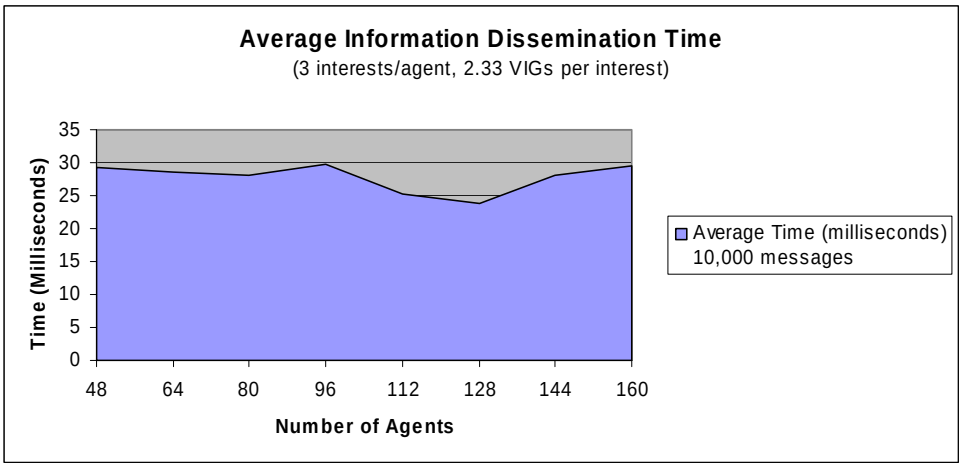


Figure 21: Average Information Dissemination Time. The time to disseminate information messages follows a constant trend against increasing number of agents and increasing volume of messages.

The graph in Figure 21 show that the average times to disseminate information over increasing number of agents and volume of information remains constant. The times are averaged over the volume of information (10000 messages). The variability in times is due to different number of VIGs generated. Each data point in the graphs is averaged over 3 runs. The constant trend is expected as the number of VIGs does not increase exponentially. For an exponential increase, the dissemination time would have followed a linear trend.

Appendix C: TID API Specification

1: ClientAgent

Handles all TID client functions.

registerClient(Client)

Adds a Client object to the observer list. The ClientAgent notifies all registered Clients when it receives a new Document.

removeClient(Client)

Removes a Client from the observer list. The Client will no longer receive new Document notifications.

publish(Document)

Publish a new document to the TID network. The ClientAgent will process the document and determine the destination VIGs.

publish(Document, Interest)

Sends a document to a specific interest group.

addTask(Task)

Adds a new task to the ClientAgent's task list. ClientAgent will join interest groups based on the Task's interests.

removeTask(Task)

Removes a task from the ClientAgent's task list. ClientAgent will unsubscribe from any VIGs that they are no longer interested in.

addInterest(Task, Interest)

removeInterest(Task, Interest)

getTaskList() -> List<Task>

Returns the list of current Tasks. The Client can then manipulate this list directly.

2: Client

Interface implemented by client software.

recieveNewMessage(Document)

Called by the associated ClientAgent when a new document is received.

sendNewMessage(Document)

sendNewMessage(Interest, Document)

3: Document

Object that contains message content and metadata. This will be used for all messages (instant messages and published documents). The Document class is written as a generic, so it can easily be used to store any type of object.

Initially, documents could be created directly, but as we get more document types and metadata, we might want to build a factory class to produce valid documents.

See Prototype.

Document(<E>)

Default Constructor. Builds a new document and sets the content and document type. The metadata starts empty.

getContents() -> <E>

Gets the Document's contents.

setContents(<E>)

Sets the Document's contents.

get(Key) -> Object

Gets the metadata associated with the given Key. If the Key's data has not been set, it returns null.

set(Key, Object)

Sets the metadata associated with the given Key. The metadata object must be an instance of the Key's associated class (see Key.validate()).

getMetaDataKeys() -> Set<Key>

Returns a Set containing all metadata Keys that contain data.

getDocumentType() -> DocumentType

Returns the document's type.

4: Key

An enum of (key, class) pairs that define the allowable metadata. Metadata is optional data about the document. This enum limits the metadata (and the class of the data stored there) to a pre-defined list. We can easily expand the valid metadata variables by adding new elements to this enum.

The system could be made more dynamic by removing the class restrictions, or by eliminating the Key enum entirely and just use arbitrary keys. However, if different client apps (potentially from different organizations) use the same TID network, then it seems important to carefully control the metadata.

See Prototype.

validate(Object) -> boolean

Returns true if the object is an instance of this Key's associated class.

5: DocumentType

An enum of (name, class) pairs that define the currently supported document types. See Prototype.

static getType(Object) -> DocumentType

Returns the DocumentType for a given object.

validate(Object) -> boolean

Returns true if the object is an instance of this DocumentType's associated class.

6: Task

Wrapper for a user task and it's collection of instances.

Task(String)

Constructor: builds a task from a natural language description of the task. It parses out the initial Interests.

addInterest(Interest)

Add a new Interest to this Task.

removeInterest(Interest)

Removes the given Interest from this Task.

getInterests() -> List<Interest>

Returns the list of all Interests for this Task.

toString() -> String

Returns the original, natural language description.

7: Interest

Represents a user's interest. Currently just a wrapper around a String, but making it it's own class would allow us to expand the interests (if needed), allow type checking and (most importantly) make the API more explicit.

Interest(String)

Constructor: builds an Interest from the given String.

toString() -> String

Returns the original String used to create this Interest.

Prototypes:

```
public enum Key {
     TYPE(DocumentType.class),
     AUTHOR(String.class),
     ORGANIZATION(String.class),
     SOURCE(String.class),
     DATE(Date.class),
     TITLE(String.class),
     INTEREST(Integer.class),
     NOVELTY(Integer.class),
     USER1(Object.class);

     private Class type;

     Key(Class type){this.type = type;}

     public boolean validate(Object object){
          return type.isInstance(object);
     }
}

-------------------------------------------------------

public enum DocumentType {
     TEXT(String.class),
     UNDEFINED(Object.class);


     private Class type;

     DocumentType(Class type){this.type = type;}

     public boolean validate(Object object){
          return type.isInstance(object);
     }

     public static DocumentType getType(Object object){

          for(DocumentType docType: DocumentType.values()){
```

```
                if(docType.validate(object))
                     return docType;
           }

           return DocumentType.UNDEFINED;
     }
}
```

```
public class Document<E> {

     private E contents;
     private DocumentType type;
     private Hashtable<Key, Object> metaData;

     public Document(E contents){

          this.contents = contents;
          metaData = new Hashtable<Key, Object>();
          type = DocumentType.getType(contents);
     }

     public E getContents(){return contents;}

     public void setContents(E contents){
          this.contents = contents;
     }

     public Object get(Key key){return metaData.get(key);}

     public void set(Key key, Object value){
               if (!key.validate(value))
                    throw new RuntimeException(value +
                         " is not a valid " + key);

               metaData.put(key, value);
          }

     public Set<Key> getMetaDataKeys(){
          return metaData.keySet();
     }

     public DocumentType getDocumentType(){
          return type;
     }

     public static void main(String[] args){
          Document<String> sdoc = new
               Document<String>("This is a test.");

          sdoc.set(Key.AUTHOR, "John X. Doe");
          sdoc.set(Key.DATE, new Date());
          sdoc.set(Key.INTEREST, 5);
```

```
            String contents = sdoc.getContents();
            System.out.println(contents +
                 " (as " + sdoc.getDocumentType() + ")");

            Set<Key> keys = sdoc.getMetaDataKeys();

            for(Key key: keys){
                 System.out.println(key +
                      ": " + sdoc.get(key));
            }
       }
}
```

Appendix D: Confusion Matrix

=== Confusion Matrix ===

```
   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o   p   q   r   s   t   <-- classified as
 264   1   0   1   1   0   0   0   2   0   0   1   0   1   1   9   1   5   2  22 |   a = alt.atheism
   1 309  15  13   7  18   1   1   0   1   0   3   6   0   1   0   0   1   0   0 |   b = comp.graphics
   1  18 323  22   3  13   1   0   0   0   1   1   1   0   1   0   0   0   0   2 |   c = comp.os.ms-windows.misc
   0  11  25 319  19   0   9   0   0   0   0   2   6   1   0   0   0   0   0   0 |   d = comp.sys.pc.hardware
   0   4   8   9 342   2  10   0   1   0   0   0   6   0   0   0   0   0   0   0 |   e = comp.sys.mac.hardware
   0  22  12   3   2 337   2   0   1   0   0   0   1   0   2   0   0   0   0   0 |   f = comp.windows.x
   0   4   2  21  13   0 309  11   3   1   2   0   8   2   1   0   1   0   0   0 |   g = misc.forsale
   0   3   0   2   0   0   8 372   3   0   1   0   3   0   1   0   2   0   0   0 |   h = rec.autos
   0   0   0   0   1   1   2   4 385   0   0   0   2   0   0   0   0   0   2   0 |   i = rec.motorcycles
   1   4   0   0   2   0   3   1   0 380   5   0   0   0   0   0   0   0   0   0 |   j = rec.sport.baseball
   2   0   2   2   0   0   0   0   0   3 382   0   0   0   0   3   0   0   0   0 |   k = rec.sport.hockey
   0   3   2   1   2   3   0   1   0   1   0 371   3   1   0   0   4   0   2   0 |   l = sci.crypt
   0   5   4  16  13   2   4   5   1   0   1   1 335   2   3   0   0   0   0   1 |   m = sci.electronics
   3   6   0   1   1   2   1   0   0   0   0   0   7 351   4   0   1   2   2   0 |   n = sci.med
   0   7   2   0   1   1   1   1   0   0   0   1   1   3 366   2   2   0   2   0 |   o = sci.space
  10   1   2   0   1   1   0   0   1   2   2   0   3   4   0 349   0   0   0   5 |   p = soc.religion.christian
   1   1   0   0   1   0   0   0   3   1   0   3   1   0   0   0 338   1   8   2 |   q = talk.politics.guns
   1   1   0   0   0   0   1   1   1   1   0   1   0   1   0   2   1 344   0   0 |   r = talk.politics.mideast
   4   2   0   0   0   0   0   1   1   0   0   3   0   2   5   0  37   2 239   1 |   s = talk.politics.misc
  40   3   2   0   0   0   0   1   0   1   0   0   0   3   1  11  18   2   9 155 |   t = talk.religion.misc
```

	Precision:	Recall:
a:	84.89%	80.49%
b:	81.96%	76.29%
c:	83.46%	80.95%
d:	85.52%	77.80%
e:	89.53%	83.62%
f:	88.22%	88.68%
g:	81.75%	87.78%
h:	94.18%	93.23%
i:	96.98%	95.77%
j:	95.96%	97.19%
k:	96.95%	96.95%
l:	94.16%	95.87%
m:	85.24%	87.47%
n:	92.12%	94.61%

o:	93.85%	94.82%
p:	96.60%	92.82%
q:	93.89%	83.46%
r:	96.90%	96.36%
s:	80.47%	89.85%
t:	63.00%	82.45%